Inside the Research C# Compiler

David R. Hanson and Todd A. Proebsting
Programming Language Systems Group
Microsoft Research
http://www.research.microsoft.com/pls/

Motivation

- C# presents an opportunity for new language features to see wide use in real applications
- Language design research is experimental—MS programmers don't buy paper designs
- .NET provides much of the required infrastructure
- Missing piece: a C# compiler that is
  - Complete
  - Flexible
  - Easy to understand
  - Easy to modify
  - Reasonably efficient

Results in a Nutshell

- Research C# Compiler aka lcsc (local Csharp compiler)
- Written in C#, grok in 1-2 weeks
- Stats:

6735 lines	lexer, parser, data structures (required)
6948	traditional semantic passes (required for "normal" compiler)
6299	optional utility passes
526	program generation tools

- Simple tools: small 100s of lines
- Complete passes, complex tools: small 1000s of lines
- Typical language extensions: 500-1000 lines, sky's the limit
- Surprise: Most usage to date is for C# tools, not language design

Design From 30,000 Feet

- Dead simple
  - Parser reads C# source, builds an Abstract Syntax Tree (AST)
  - Subsequent passes traverse/modify/annotate AST
- Loosely coupled
  - Passes specified at execution time by command-line options
  - No fixed passes, no fixed order
  - No specified output or output format
- Tool-Driven
  - Parser generator
  - Generator for pass skeletons
  - Strongly typed list generator (poor man's generics)

Parsing: Source to AST

- Generalized LR parser generated automatically from possibly ambiguous C# grammar
- Parser
  - Accepts C# source
  - Builds parse trees (>1 if ambiguous)
  - Builds AST from selected parse tree using semantic actions to build AST nodes
- Grammar
  - Follows published C# grammar closely
  - Specified in Excel spreadsheet (Demo)
  - Excel formulas help get types correct
- Hand-written lexer

Microsoft Excel - [illegible]

	A	B	C	
185	variable-initializer	array-initializer	variable_initializer	a1
186	local-constant-declaration	const type constant-declarators	statement	new const_statement
187	constant-declarators	constant-declarator	List	declaration_list.Add(a1)
188	constant-declarators	constant-declarators , constant-declarator	List	List.Concat(a1,a3)
189	constant-declarator	identifier = constant-expression	const_declarator	new const_declarator
190	expression-statement	statement-expression	statement	new expression_stat
191	statement-expression	invocation-expression	expression	a1.ancestor(false)
192	statement-expression	object-creation-expression	expression	a1.ancestor(false)
193	statement-expression	assignment	expression	a1.ancestor(false)
194	statement-expression	post-increment-expression	expression	a1.ancestor(false)
195	statement-expression	post-decrement-expression	expression	a1.ancestor(false)
196	statement-expression	pre-increment-expression	expression	a1.ancestor(false)
197	statement-expression	pre-decrement-expression	expression	a1.ancestor(false)
198	selection-statement	if-statement	statement	
199	selection-statement	switch-statement	statement	
200	if-statement	if (boolean-expression) embedded-statement	statement	new if_statement(a3
201	if-statement	if (boolean-expression) embedded-statement else	statement	new if_statement(a3
202	switch-statement	switch (expression) switch-block	statement	new switch_statement
203	switch-block	{ switch-sections[opt] }	List	a2
204	switch-sections	switch-section	List	switch_section_list.Add
205	switch-sections	switch-sections switch-section	List	List.Concat(a1,a2)
206	switch-section	switch-labels statement-list	switch_section	new switch_section(a
207	switch-labels	switch-label	List	switch_label_list.Add
208	switch-labels	switch-labels switch-label	List	List.Concat(a1,a2)
209	switch-label	case constant-expression :	switch_label	new switch_expression
210	switch-label	default :	switch_label	new switch_default
211	iteration-statement	while-statement	statement	

Apply with Changes...

Add ▾ Statement

grammar / Sheet1

Microsoft Excel - Mono-plan.xls [Read-Only]

File Edit View Insert Format Tools Data Window Help Acrobat

Reply with Changes...

A2:

	A	B	C	
185	variable-initializer	array-initializer	variable_initializer	a1
186	local-constant-declaration	const type constant-declarators	statement	new const_statement
187	constant-declarators	constant-declarator	List	declaration_list(new a1
188	constant-declarators	constant-declarators , constant-declarator	List	List.Concat(a1, a2)
189	constant-declarator	identifier = constant-expression	const_declarator	new const_declarator
190	expression-statement	statement-expression ;	statement	new expression_state
191	statement-expression	invocation-expression	expression	a1.annotate(false)
192	statement-expression	object-creation-expression	expression	a1.annotate(false)
193	statement-expression	assignment	expression	a1.annotate(false)
194	statement-expression	post-increment-expression	expression	a1.annotate(false)
195	statement-expression	post-decrement-expression	expression	a1.annotate(false)
196	statement-expression	pre-increment-expression	expression	a1.annotate(false)
197	statement-expression	pre-decrement-expression	expression	a1.annotate(false)
198	selection-statement	if-statement	statement	
199	selection-statement	switch-statement	statement	
200	if-statement	if (boolean-expression) embedded-statement	statement	new if_statement(a1, a2
201	if-statement	if (boolean-expression) embedded-statement else	statement	new if_statement(a1, a2
202	switch-statement	switch (expression) switch-block	statement	new switch_statement
203	switch-block	{ switch-sections[opt] }	List	a2
204	switch-sections	switch-section	List	switch_section(List(new
205	switch-sections	switch-sections switch-section	List	List.Concat(a1, a2)
206	switch-section	switch-labels statement-list	switch_section	new switch_section(a
207	switch-labels	switch-label	List	switch_label(List(new
208	switch-labels	switch-labels switch-label	List	List.Concat(a1, a2)
209	switch-label	case constant-expression :	switch_label	new switch_expression
210	switch-label	default :	switch_label	new switch_default()
211	iteration-statement	while-statement	statement	

grammar

Microsoft Excel - CSharp-Grammar.xls [Read-Only]

File Edit View Insert Format Tools Data Window Help Adobe

Reply with Changes...

A2:

	A	B	C	
450	attribute-name	type-name	type	
451	attribute-arguments	(positional-argument-list)	attribute_arguments	new attribute_argume
452	attribute-arguments	(positional-argument-list , named-argument-list)	attribute_arguments	new attribute_argume
453	attribute-arguments	(named-argument-list)	attribute_arguments	new attribute_argume
154	positional-argument-list	positional-argument	List	expressionList.Ins
155	positional-argument-list	positional-argument-list , positional-argument	List	List.Concat(a)
156	positional-argument	attribute-argument-expression	expression	
157	named-argument-list	named-argument	List	named_argumentList
158	named-argument-list	named-argument-list , named-argument	List	List.Concat(a)
159	named-argument	identifier = attribute-argument-expression	named_argument	new named_argumen
160	attribute-argument-expression	conditional-expression	expression	
161	type	pointer-type	type	
162	pointer-type	unmanaged-type *	type	new pointer_type(a)
163	pointer-type	void *	type	new void_pointer_typ
164	unmanaged-type	type	type	
165	primary-expression	pointer-member-access	expression	
166	unary-expression	addressof-expression	expression	
167	pointer-member-access	primary-expression -> identifier	expression	new pointer_access
168	addressof-expression	& unary-expression	expression	new unary_expressio
169	sizeof-expression	sizeof (unmanaged-type)	expression	new sizeof_expressio
170	unsafe-statement	fixed-statement	statement	
171	fixed-statement	fixed (pointer-type fixed-pointer-declarators) em	statement	new fixed_statement
172	fixed-pointer-declarators	fixed-pointer-declarator	List	declaratorList.Ins
173	fixed-pointer-declarators	fixed-pointer-declarators , fixed-pointer-declarator	List	List.Concat(a)
174	fixed-pointer-declarator	identifier = expression	fixed_declarator	new fixed_declarator
175	variable-initializer	stackalloc-initializer	variable_initializer	
176	variable-initializer	stackalloc unmanaged-type [expression]	variable_initializer	new stackalloc_initia
177				

grammar / Sheet1 /

Ready

Parsing: Source to AST

- Generalized LR parser generated automatically from possibly ambiguous C# grammar
- Parser
  - Accepts C# source
  - Builds parse trees (>1 if ambiguous)
  - Builds AST from selected parse tree using semantic actions to build AST nodes
- Grammar
  - Follows published C# grammar closely
  - Specified in Excel spreadsheet (Demo)
  - Excel formulas help get types correct
- Hand-written lexer

Abstract Syntax Trees

- 164 node types; 16 are abstract
- Follows C# syntax closely; strongly typed

```
public class if_statement: statement {
    public if_statement(expression expr,
            statement thenpart, statement elsepart) {
        this.expr = expr;
        this.thenpart = thenpart;
        this.elsepart = elsepart;
    }
    public expression expr;
    public statement thenpart;
    [MayBeNull] public statement elsepart;
    public override void visit(ASTVisitor prefix,
            ASTVisitor postfix) { ... }
}
```

- Lists are strongly typed [Demo]

```
while declarationlist body;
```

Abstract Syntax Trees

```
public class if_statement : statement {
    public if_statement(expression expr
            statement thenpart : statement elsepart) {
        this.expr = expr;
        this.thenpart = thenpart
        this.elsepart = elsepart
    }
    public expression expr
    public statement thenpart
    [MayBeNull] public statement elsepart
    public override void visit(ASTVisitor prefix
            ASTVisitor postfix) {    }
}
```

Semantics

```
public class if_statement  statement {

    public override void visit(ASTVisitor prefix
            ASTVisitor postfix) {
        prefix(this)
        expr visit(prefix  postfix)
        thenpart visit(prefix  postfix)
        if (elsepart != null)
            elsepart visit(prefix  postfix)
        postfix(this);
    }
}
```

S mple C# Tools

```
public static void prefix() {    }
public static void postfix() {    }
public static void Main(string[] args) {
    foreach (string s in args) {
        AST ast = (AST)Parser parse(s,
                                    new StreamReader(s));
        if (ast != null)
            ast visit(new ASTVisitor(prefix),
                      new ASTVisitor(postfix)).
    }
}
```

Finding Assertions

```
if (..) throw

public static void doit(AST ast) {
  if (ast is if_statement
  && ((if_statement)ast) thenpart is throw_statement
  && ((if_statement)ast) elsepart == null)
      Console.WriteLine("{0}  Possible assertion",
          ast.begin);
}

public static void Main(string[] args) {

  ast visit(new ASTVisitor(doit))
}
```

```
/find-assert
/mkvisitor -class bind -arg
/lose visitor XML hello cs
  x  users dsh sshsrp compiler inside sh .Shell Fill, Top
```

```
(process) [Process] End *
```

```
/find-assert  ls * | grep -v -e tables -e
/mkvisitor -class bind -arg  -pkg  Tab e t  dirge  >foo 2>&
 lose  visitor XML hello 2>&
  x  users drh csharp compiler Inside sh  Shell Fill  Top
X  users drh csharp compile  find-assert  ls * 2>&1 grep -v -e tables 2>&

(process) [Process] End *
```

```
find-assert [illegible] grep [illegible] -e tables [illegible]
/mkvisitor -class bind -arge [illegible] >foo.cs
close visitor XML hello.cs
x users dsh sharp compiler inside sh (Shell Fill) Top
x [illegible] find-assert [illegible] * [illegible] grep -v -e tables cs
Types.cs 55 5) possible assertion
disambiguate cs(9 4) Possible assertion
|
```

(process) [Process] End *

```
/find-assert .s * | grep -v -e tables -e
/mkvisitor -class bind -arge [illegible] >foo.cs
/lese visitor XML hello.cs
  x  users drh .sharp compiler Inside sh (Shell fill) Top
X /users/drh .sharp comp [illegible] /find-assert ls * cs grep -v -e tables cs
Types.cs(55,4)  possible assertion
disambiguate.cs(9,4)  Possible assertion
[illegible]

  (process) [Process] End *
```

Finding Assertions

```
if (..) throw

public static void doit(AST ast) {
  if (ast is if_statement
  && ((if_statement)ast) thenpart is throw_statement
  && ((if_statement)ast) elsepart == null)
      Console.WriteLine("{0}: Possible assertion",
          ast.begin);
}

public static void Main(string[] args) {

  ast.visit(new ASTVisitor(doit));
}
```

Lame

Visitors

```
public static object visit(object ast, TextWriter w)
```

- [illegible]
- Emit output of any to w

lcsc -visitor

- [illegible]
- [illegible]

  mkvis for -class bind -args SymbolTable bindings >foo.cs
- Demo

```
mBile declarationlist body/
```

SymbolTable binding!

```
ast.elsequart t- nalll
```

Visitors

```
public static object Visit(object ast, TextWriter w)
```

- [illegible]
- Emit output if any to w

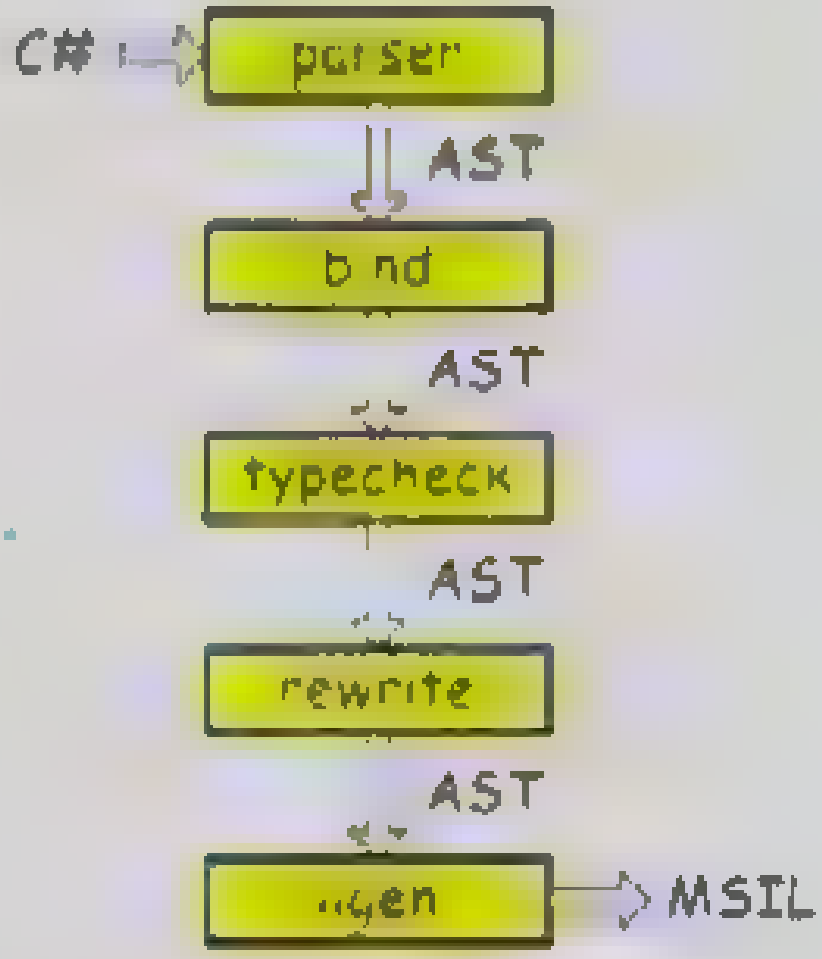
lcsc -visitor

- [illegible]
- [illegible]

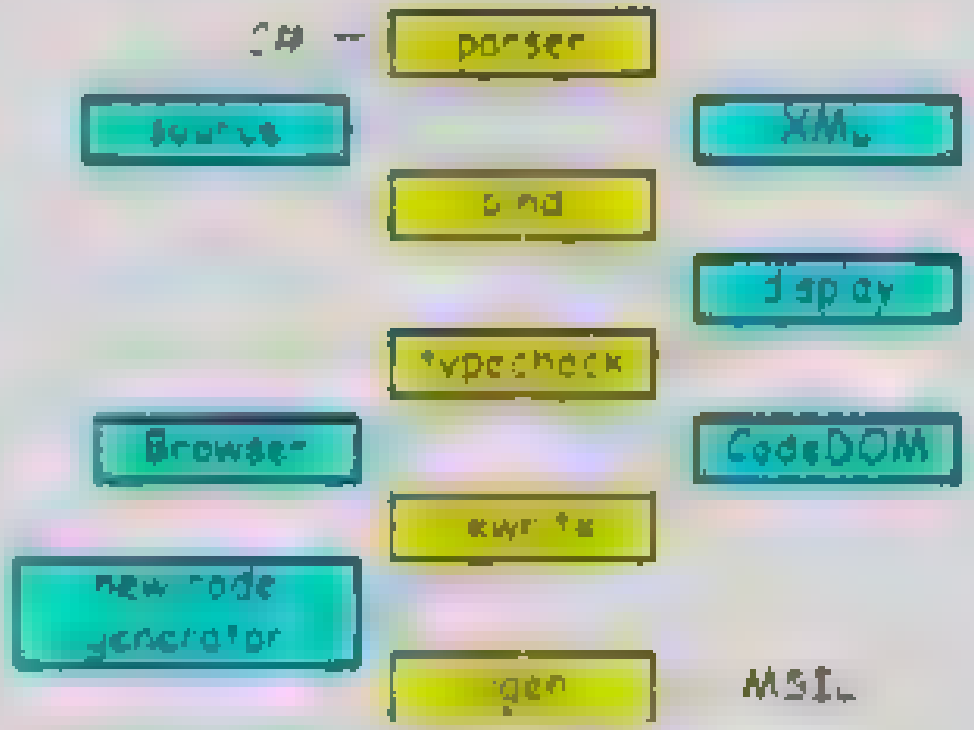
  mkvis for -class bind -args SymbolTable bindings >foo.cs
- Demo.

Traditional Compiler Passes

- Parser
  - annotate with Symbols
- typecheck
  - Method
- rewrite
  - generation
- codegen
  - emit MSIL



"Exotic" Visitors



Visitor Examples

- [illegible] Demo

  lcsc -visitor XML hello.cs

- [illegible] Demo

  lcsc -visitor source hello.cs

- [illegible] Demo

  lcsc -visitor sortmembers -visitor source mkvisitor.cs

- [illegible] Demo

  lcsc -visitor bind -visitor display debug/base.dll hello.cs

```
/find-assert  [illegible] grep [illegible] [illegible]
/mkvisitor -class bind -arg [illegible] bindings >foo.cc
 lese -visitor XML hello.cc
  x  users dsh zsharp compiler Inside sh (Shell Fill) Top
[illegible]  find-assert [illegible] * cc grep -v -e tabled.cc
Types.cc 55 5)  Possible assertion
disambiguate.cc(9 4)  Possible assertion
[illegible]

(process) [Process] End *
```

```
./find-assert  x  x grep  a tables x
/mkvisitor -class bind -arg  SymbolTab * bindings  >foo cs
/lcsc -visitor XML hello cs
  x  users drh csharp compiler Inside sh [Shell Fill] Top
                  <tag>identifier< tag>
              </InputElement>
            </id>
            <sym/>
            <mods>
            < mods>
          <end>hello cs 0 2)</end>
        </class_declaration>
        < declarations<
        <using_directives>
        </using_directives>
        <sym/>
      <end>hello cs(0 2)< end>
    </compilation_unit>
    </inputs>
    <assemblyRefs/>
  </compilation>
  <end>(0 0)</end>
</program>
[XML: 0 0200s]
[Total: 0 6510s]
X  users drh csharp compiler 2
   (process  [Process, End *
```

```
,mkvisitor -class bind -arg  2,mk 1Table bindings  >foo.cs
/lose -visitor XMI hello cs
/lose  visitor source hello cs
  x  users drh csharp compiler Inside sh ,Shell Fill, 20>
        </using_directives>
        <sym/>
      <end>hello cs(0 2)< end>
    </compilation_units>
    </inputs>
    <assemblyRefs/>
  </compilation>
  <end>(0.0)</end>
< program>
,XMI, 0 0200s,
[Total  0 6810s]
                                    lose -visitor source hello cs
[parse  0 6109s]

class Hello {
        public static void Main()
                System Console WriteLine( Hello  world )
        |

[source: 0 0300s]
[Total, 0 6810s]
X /users drh csharp compiler 2
    process  [Process, End *
```

Visitor Examples

[illegible] Demo

```
lcsc -visitor XML hello.cs
```

[illegible] Demo

```
lcsc -visitor source hello.cs
```

[illegible] Demo

```
lcsc -visitor sortmembers -visitor source mkvisitor.cs
```

[illegible] Demo

```
lcsc -visitor bind -visitor display debug/base.dll hello.cs
```

3 7 2003 11 25 52 AM

compilation#7

    origstringList(+visitor bind +visitor dep by debug base dll helio cs)

    inputs=[illegible]

    assemblyRefs=stringList()

    parent=null

compilation_unit#10

    [illegible]

compilation_unit#13

    attributes=attribute_section_list#15 (empty),

    declarations=declaration_list#17

    using_directives=using_directive_list#19 (empty)

    symbol=[illegible]

    parent=[illegible]

declaration_list#17

    class_declaration#23

NameSpace#21

    aliases=stringList()

    usingdirectives=null

```
class declaration#23
    attrs=attribute_section_list#33 (empty)
    bases=type_list#35 (empty)
    body=[illegible]#37
    id=InputElement(column=[illegible], str=[illegible])
    symbol=[illegible]#4[illegible]
    modifiers=[illegible]#45 (empty)
    parent=[illegible]
Symbol Table#27
    Owner=[illegible]#[illegible]
    {System.}NameSpace#40
    {[illegible]}[illegible]#4[illegible]
    {Microsoft}=[illegible]
declaration_list#37
    method_declaration#51
    [illegible]#[illegible]
    [illegible]
ClassType#43
    baseClass=[illegible]
    interfaces=InterfaceTypeList#57 (empty
```

declarat on_ st#37

methoq_ 3s_ ıarat on#5)

ClassType#43

baseClass=

interfaces=[interfaceType_ st#57 empty)

members=

enclosingType=null

module=null

attributes=null

dec Space=

id=InputElement(coord=he ıo cs(3 7 st=we io tag= dent f er)

mod f ers=str ngL st(internal)

serial=59

NameSpace#50

aliases=strıngL st()

usıngdıect ves=nu

members=

module=null

Method#79

interfaceMethod=null

locals=symbolTable#127 (empty)

signature=[illegible]

module=null

attributes=null

decSpecs=[illegible]

id=InputElement(coord=hello.cs(4,21) str=Main tag=identifier)

modifiers=string_list(public static)

serial=61

InputElementList#81

InputElement(coord=hello.cs(4,2) str=public tag=public)

InputElement(coord=hello.cs(4,9) str=static tag=static)

block_statement#86

stmts=statementList#136 (empty)

symbolTable=[illegible]

method=Method#90

lab=0

parents=[illegible]

constructor_declaration#88

```
class_declaration#23
    attrs=attribute_sectionList#33 empty,
    bases=typeList#34 empty,
    body=[illegible]#37
    isInputElement=[illegible]
    symbolType#4
    modifiedInputElementsList#45 empty)
    parent=[illegible]
SymbolTable#27
    Owners=[illegible]#2
    {System}=NameSpace#40
    {[illegible]}=[illegible]#4
    {Microsoft}=[illegible]#5,
declarationList#37
    method_declaration#51
    [illegible]#[illegible]
    [illegible]#[illegible]
ClassType#43
    baseClass=[illegible]#[illegible]
    interfaces=InterfaceTypeList#57 empty
```

Navigating C# Source and ASTs

- [illegible]
- [illegible]

```
lcsc -visitor Browser BrowserVisitor debug/Browser dll hello cs
```

```
lcsc -visitor bind -visitor typecheck
   -visitor Browser BrowserVisitor debug/Browser dll Bq cs
```

SourceObject Browser

Back | Forward | Parent | Add Object Browser | Add CodeDom Browser | Add [illegible] Browser

Sid hello CSR1 2002 03 08 11 20 12 REDMOND [illegible] compilation_unit compilation_unit 176

```
class Hello {
    [EntryPoint] void Main() {
        System.Console.WriteLine("hello, world");
    }
}
```

hello cs

Object Browser

Back | Forward | Parent | Add Object Browser | Add CodeDom Browser | Add [illegible] Browser

Sid [illegible] CS#1 2002 03 08 [illegible] REDMOND [illegible]

```
class Hello {
    public static void Main()
        System.Console.WriteLine("Hello, world");
}
```

(compilation_unit) compilation_unit#175

attributes attribute_section_s#1 0 empty

declarations declaration_s#183

using_directives using_directive_s#165 e

symbol

parent compilation#100

get_end <property>

get_begin <property>

hello cs

Source/Object Browser

Back | Forward | Parent | Add Object Browser | Add CodeDom Browser | Add L Browser

Sid Hello.cs#1 2002-03-08 [illegible] PEDNIONO

```
class Hello {
    public static void Main [illegible]
        System.Console.WriteLine("Hello World
    }
}
```

```
compilation get_compilation_unit#176
attributes attribute_section_list#179 empty
declarations declaration_list#180
using_directives using_directive_list#185 e
symbols
parent compilation#190
get_end_property,
get_begin_property,
(declarations.cs) declarations.cs#184
```

hello.cs

Source/Object Browser

Back | Forward | Parent | Add Object Browser | Add CodeDom Browser | Add L Browser

3id hello.cs#1 2002.03.08 14:30:12 REDMOND

```
class Hello {
    [STAThread] void Main() {
        System.Console.WriteLine("hello, world");
    }
}
```

```
compilation_unit compilation_unit#176
  attributes attribute_section_list#179 empty
  declarations declaration_list#183
  using_directives using_directive_list#185 em
  sym null
  parent compilation#190
  get_and <property>
  get_begin <property>
  declaration_list declaration_list#182
  get_Count <property>
  get_inner_list <property>
  get_list <property>
  [illegible]
(class declaration) class declaration#230
```

hello cs

Back | Forward | Parent | Add Object Browser | Add CodeDom Browser | Add [illegible] Browser

```
Sid Hello csx1 2002 03 08 1[illegible] 30 [illegible] REDMOND [illegible]

class Hello {
    public static void Main() {
        System.Console.WriteLine("Hello, world")
}
```

```
compilation_unit: compilation_unit#176
  attributes: attribute_section_list#179 (empty)
  declarations: declaration_list#182
  using_directives: using_directive_list#185 [illegible]
  symbol: [illegible]
  parent: compilation#190
  get_end: <property>
  get_begin: <property>
declaration_list: declaration_list#182
  get_Count: <property>
  get_InnerList: <property>
  get_List: <property>
  [illegible]: class_declaration#230
class_declaration: class_declaration#230
(string_literal) string_literal#272
  token: [illegible]IdentifierElement: coorg=hello.cs:8,28 str[illegible]
  valueUsed: True
  type: null
  value: null
  parent: argument#283
  get_end: <property>
  get_begin: <property>
```

hello.cs

Source/Object Browser

Back | Forward | Parent | Add Object Browser | Add CodeDom Browser | Add [illegible] Browser

```
3id "Hello.cs#1" 2002-03-08 11:30:12 REDMOND [illegible]

class Hello {
    public static void Main() {
        System.Console.WriteLine("Hello, world");
    }
}
```

```
compilation_unit compilation_unit#176
attributes attribute_section_list#179 empty
declarations declaration_list#182
using_directives using_directive_list#185 em
symbol null
parent compilation#190
get_end <property>
get_begin <property>
declaration_list declaration_list#182
get_Count <property>
get_InnerList <property>
get_List <property>
[illegible] class_declaration#230
class_declaration class_declaration#230
[illegible]
token_input_Element:Coordinates 0.05.5.28 str
resolved_type True
type null
value null
parent argument#281
get_end <property>
get_begin <property>
```

hello.cs

Back | Forward | Parent | Add Object Browser | Add CodeDom Browser | Add Browser

```
class Hello {
    public static void Main() {
        System.Console.WriteLine("Hello, world")
```

```
declarations: declaration_list#182
using_directives: using_directive_list#185
sym: null
parent: compilation#100
get_end: <property>
get_begin: <property>
declaration_list: declaration_list#182
get_Count: <property>
get_InnerList: <property>
get_List: <property>
class_decl: class_decl#230
class_decl: class_decl#230
string_lit: string_lit#232
location: input:Element(coordinates 5,28)
isUsed: True
type: null
value: null
parent: argument#233
get_end: <property>
get_begin: <property>
argument: argument#233
```

hello.cs

Navigating C# Source and ASTs

- [illegible]
- [illegible]

```
lcsc -visitor Browser BrowserVisitor debug/Browser dll hello cs
```

```
lcsc -visitor bind -visitor typecheck
   -visitor Browser BrowserVisitor debug/Browser dll Bg cs
```

Source/Object Browser

Back | Forward | Parent | Add Object Browser | Add CodeDom Browser | Add Browser

```
class EightQueens {
    [illegible] = new bool[15], down = new bool[15]
    [illegible] {
        [illegible] [illegible]
        [illegible]
        [illegible] [illegible]
        [illegible]
        [illegible] new int[8]
    }
    [illegible] void [illegible] int [illegible] {
        [illegible]
        [illegible] {
            [illegible] = true
            [illegible]
            [illegible]
            [illegible] [illegible]
            [illegible] = true
        }
        [illegible] void [illegible] {
            [illegible] (int i = 0; i < 8; i)
```

```
[illegible] Object
    [illegible]
    [illegible]
    [illegible] void Main
    [illegible] (string[] [illegible])
    [illegible] Console.[illegible]
    [illegible] //
    [illegible]
    [illegible]
    [illegible] = true
        [illegible] EightQueens.up = true
    [illegible]
        [illegible] EightQueens.down
    [illegible]
        [illegible] true
        [illegible]
        [illegible] true
        [illegible]
    [illegible]
    [illegible] 0 //
```

8a cs | 8a cs

Back | Power d | Parent | Add Object Browser | Add CodeDom Browser | Add IL Browser

```
        up[i] = down[i] = true
    for int i = 0; i < rows.Length; i++
        rows[i] = true
    queens[0] = new int[8];

static void PlaceQueens(int[] queens, int r, int[] up, int[] down, int[] rows)
    for int c = 0; c < rows.Length; c++
        if (rows[c] && up[r + c] && down[r - c])
            rows[c] = up[r + c] = down[r - c] = false
            queens[r] = c
            if (r == 7)
                PrintArray(queens);
            else
                PlaceQueens(r + 1, queens, up, down, rows);
            rows[c] = up[r + c] = down[r - c] = true
        }
}
static void PrintArray(int[] x) {
    foreach (int c in x)
        System.Console.Write("{0}", c + 1);
    System.Console.WriteLine();
}
```

```
$11:
ret
.maxstack 12
    end of method EightQueens::Queens
.method public hidebysig static private void PrintArray(
.locals init ([0]int32 c)
  IL_0000: ldarg.0
.locals init ([1]int32 V_1)
  IL_0000: ldc.i4.0, int32, int32, int32
ldarg.0    x
stloc.2    t2
ldc.i4.0
stloc.1    i1
br.s $20
$17:
  IL_0005: ldarg.0    x
ldloc.2    t2
ldloc.1    i1
ldelem.i4
stloc.0    c
  IL_0008: ldc.i4.4   System.Console::Write("{0}"
IL_000f:
ldloc.0    c
```

Ready | Ready

Back | Forward | Parent | Add Object Browser | Add CodeDom Browser | Add IL Browser

```
        up[i] = down[i] = true
    for int i = 0 ... rows.Length ...
        rows[i] = true
    queens[0] = new int[8];

static void queens(int[] x) {
    for int r = 0 ... rows.Length ...
        if (rows[r] && up[r+c] && down[r-c]) {
            rows[r] = up[r+c] = down[r-c] = false;
            x[c] = r;
            if (c == 7)
                print(x);
            else
                queens(c + 1, x);
            rows[r] = up[r+c] = down[r-c] = true;
        }
}

static void print(int[] x) {
    foreach (int c in x)
        System.Console.Write("{0}", c + 1);
    System.Console.WriteLine();
}
}
```

Bq.cs

```
.locals init ([0]int32 c)
  Bc.cs 23.21: x
.locals init ([1]int32 [1])
.locals init ([2]int32[,] [2])
ldarg.0   x
stloc.2   [2]
ldc.i4.0
stloc.1   [1]
br.s IL_20
ldarg.1
  Bq.cs 23.3: c in x
stloc.2   [2]
stloc.1   [1]
ldelem.i4
stloc.0   c
  Bc.cs 24.4: System.Console.Write("{0}"
ldstr "{0}"
ldloc.0   c
ldc.i4.1
add
box [mscorlib]
call void [mscorlib]System.Console::Write(
IL_5b:
```

Bq.cs

Source/Object Browser

Back | Forward | Parent | Add Object Browser | Add CodeDom Browser | Add [illegible] Browser

```
        up[i] = down[i] = true
    for int i = 0 ... rows.Length ...
        rows[i] = true
    queens = new int[8];

static void queens(int[] x)
    for int r = 0; r < rows.Length ...
        if (rows[r] && up[r+c-7] && down[r-c+7]) ...
            rows[r] = up[r+c-7] = down[r-c+7] = false
            x[c] = r
            if (c == 7)
                print(x);
            else
                queens(c + 1, x);
            rows[r] = up[r+c-7] = down[r-c+7] = true
        }

static void print(int[] x) {
    foreach (int c in x)
        System.Console.Write("{0} ", c + 1);
    System.Console.WriteLine();
}
}
```

Bg cs

```
EightQueens::rows
EightQueens::up
EightQueens::down
```

Bg cs

Navigating C# Source and ASTs

- [illegible]
- [illegible]

```
lcsc -visitor Browser BrowserVisitor debug/Browser.dll hello.cs
```

```
lcsc -visitor bind -visitor typecheck
   -visitor Browser BrowserVisitor debug/Browser.dll Bq.cs
```

Extending C# Adding typeswitch

```
typeswitch (o) {
   case Int32 (x)  Console.WriteLine(x); break;
   case Symbol (s)  Symbols.Add(s)  break
   case Segment  popSegment()  break
   default  throw new ArgumentException()
}
```

Downcasts

```
if (o is Int32) {
   int x = (int)o  Console.WriteLine(x)  }
else if (o is Symbol) {
   Symbol s = (Symbol)o  Symbols.Add(s)  }
else if (o is Segment) { popSegment()  }
else { throw new InvalidArgument() }
```

The Easy Way: Source-to-Source Transformation

- [illegible]
- [illegible]
- [illegible]
  - [illegible]
  - [illegible]

```
ent.eisopart != null)
```

```
/lose -visitor XML hello cs
/lose -visitor source hello cs
 lose -visitor sortmembers -visitor source mkvisitor cs >bas cs
  x:\users dzh sharp compiler inside sh .Shell Fill, 25>
        </using_directives>
        >sym/>
      •end•hello cs(0 2)</ end>
    </compilation_unit>
    </inputs>
    •assemblyRefs/>
  </compilation>
  <end>(0.0)</end>
</program>
.XML- 0 0200s,
[Total: 0.6810s]
                                    lose -visitor source hello cs
[parse. 0.6109s]

class Hello {
        public static void Main() {
                System Console WriteLine( Hello  world )
        }
}
[source: 0.0300s]
[Total: 0 6810s]
x  users dzh ^sharp
   (process  [Process, End *
```

```
/lcsc -visitor bind -visitor display debug base dll hello cs
/lcsc -visitor Browser BrowserVisitor debug Browser dll hello cs
/lcsc -visitor bind -visitor typecheck -visitor Browser BrowserVisitor debu
  x  users drh csharp compiler inside sh .Shell fill, 34:
        < using_directives>
        <sym/>
      <end>hello cs(0 2)< end>
    < compilation_unit>
    </inputs>
    <assemblyRefs/>
  </compilation>
  <end>(0.0)</end>
</program>
.XML: 0 0200s.
[Total: 0.6810s]
/ -v -visitor source hello cs
[parse: 0.6109s]

class Hello {
        public static void Main() {
                System Console WriteLine( Hello  world )
        }
}
[source: 0.0300s]
[Total: 0.6810s]
%  users drh csharp compiler 2
    process) [Process] End *
```

```
/lose -visitor Browser BrowserVisitor debug Browser dll hello cs
/lose -visitor bind -visitor typecheck -visitor Browser BrowserVisitor debu
 lose -visitor typeswitch source demos typeswitch sample cs
  %  users dsh csharp compiler Inside sh .Shell Fill, 65-
        < using_directives>
        <sym/>
      <end>hello cs(0 2)</end>
    < compilation_unit>
    </inputs>
    <assemblyRefs/>
  </compilation>
  <end>(0.0)</end>
</program>
.XML- 0 0200s,
[Total: 0.6910s]
                                        lose -visitor source hello cs
[parse: 0.6109s]

class Hello {
        public static void Main() {
                System Console WriteLine( Hello  world )
        }
}
[source: 0.0300s]
[Total: 0 6910s]
%  users dsh csharp compiler 2
   process, [Process] End *
```

```
/lose -visitor Browser BrowserVisitor debug/Browser dll hello cs
/lose -visitor bind -visitor typecheck -visitor Browser BrowserVisitor debu
/lose -visitor typeswitch source demos typeswitch sample cs
  x  users dsh csharp compiler inside sh .Shell Fill, cs>
        < using_directives>
        <sym/>
      <end>hello cs(0 2)< end>
    </compilation_unit>
    </inputs>
    <assemblyRefs/>
  </compilation>
  <end>(0.0)</end>
</program>
.XML: 0 0200s,
[Total: 0.6810s]
                                    lsv -verbose source hello cs
[parse. 0.0109s]

class Hello {
        public static void Main() {
                System Console WriteLine( Hello  world )
        }
}
[source: 0.0300s]
[Total: 0 6810s]
                                    |
    process) .Process] End *
```

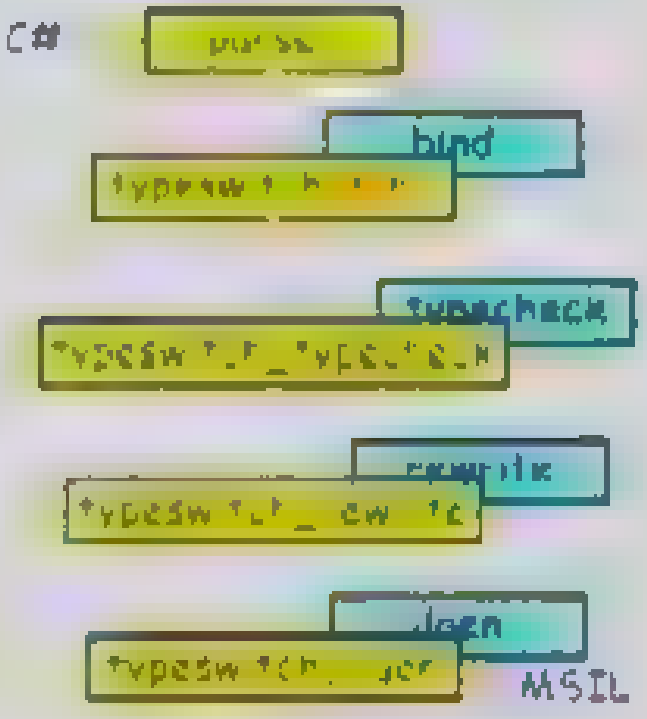
```
lcsc -visitor Browser BrowserVisitor debug Browser dll hello cs
lcsc -visitor bind -visitor typecheck -visitor Browser BrowserVisitor debu
lcsc -visitor typeswitch source demos typeswitch sample cs
  x  users drh csharp compiler inside sh .Shell fill, cs'
                        if (yy_1 is Int32) {
                                int32 x = (int32)yy_1
                                Console.WriteLine(x)
                                goto yy_1_end
                        }
                        if (yy_1 is Symbol) {
                                Symbol s = (Symbol)yy_1
                                Symbols.Add(s)
                                goto yy_1_end
                        }
                        if (yy_1 is Segment) {
                                yyySegment()
                                goto yy_1_end
                        }
                        throw new ArgumentException()
                yy_1_end:
                {
        }
}
[typeswitch_source  0.0501s]
[Total: 1.3319s]
  (process, 'Process' End *
```

The Easy Way: Source-to-Source Transformation

- Preprocessor [illegible]
- Alternative
  - [illegible]
  - [illegible]

The Ful 9 Yards Emitting MSIL

- Extend existing CT ne
  emp
- Generate
  - [illegible]
  - [illegible]



```
lose -visitor Browser BrowserVisitor debug Browser dll hello cs
lose -visitor bind -visitor typecheck -visitor Browser BrowserVisitor debu
lose -visitor typeswitch source demos typeswitch sample cs
x \users dsh asharp compiler Inside sh (Shell Fill) 60%
        static ArrayList symbols
        public static void pop(object o) {
                {
                        object yy_1 = o
                        if (yy_1 is Int32) {
                                Int32 x = (Int32)yy_1
                                Console.WriteLine(x)
                                goto yy_1_end

                        if (yy_1 is Symbol) {
                                Symbol s = (Symbol)yy_1
                                symbols.Add(s)
                                goto yy_1_end
                        }
                        if (yy_1 is Segment) {
                                popSegment();
                                goto yy_1_end
                        }
                        throw new ArgumentException()
                yy_1_end: ;
                }
        }
(process) 'Process] 99% *
```

```
/lose -visitor Browser BrowserVisitor debug Browser dll hello cs
/lose -visitor bind -visitor typecheck -visitor Browser BrowserVisitor debu
 lose -visitor typeswitch source demos typeswitch sample cs
   x  users drh csharp compiler inside sh .Shell fill, 65'
call instance void [mscorlib]System.Object .ctor()
ret
 maxstack 1
} //  end of method Test Test
 method hidebysig specialname rtspecialname static private void  cctor() ,
ret
 maxstack 0
} // end of method Test Test
} // end of class Test
|
   x  users drh csharp compiler typeswitch-sample ..  Fundamental) End
```

```
/lose -visitor Browser BrowserVisitor debug Browser dll hello cs
/lose -visitor bind -visitor typecheck -visitor Browser BrowserVisitor debu
 lose -visitor typeswitch source demos typeswitch sample cs
   x  users dzh csharp compiler Inside sh (Shell fill, 65+
// demos typeswitch-sample cs 1: 42)  break
br $3
$5:
// demos/typeswitch-sample cs(12 3)  case Symbol (s)
ldloc 0 // t0
isinst class  Test/Symbol
 locals init ([2]class  Test/Symbol    t2 )
stloc 2 // t2
ldloc 2 // t2
brfalse $7
ldloc 2 // t2
stloc 0 // s
// demos/typeswitch-sample cs(12 20)  Symbols Add(s)
ldsfld class [mscorlib]System Collections ArrayList  Test    Symbols
ldloc 0 // s
callvirt instance int32 [mscorlib]System Collections ArrayList  Add (object
pop
// demos/typeswitch-sample cs(13 16)  break
br $3
$7:
// demos/typeswitch-sample.cs(13 3)  case Segment
ldloc 0 // t0
   x  users dzh csharp compiler typeswitch-sample .. 'Fundamental) 62+
```

```
/lose -visitor Browser BrowserVisitor debug Browser dll hello cs
/lose -visitor bind -visitor typecheck -visitor Browser BrowserVisitor debu
 lose -visitor typeswitch source demos typeswitch-sample cs
  x  users dzh csharp compiler inside sh .Shell Fill, 65'
maxstack 8
, /  end of method test popsegment
field static private class [mscorlib]System Collections ArrayList  Symbols
method hidebysig public static void  Foo  object  o ) {
// demos/typeswitch-sample cs(20 3)  typeswitch  o)
 locals init ( [0]object  t0 )
ldarg 0 // o
stloc 0 // t0
/  demos/typeswitch-sample cs(11 3)  case int32 (x)
ldloc 0 // t0
isinst int32
 locals init ([1]int32  t1 )
stloc 1 // t1
ldloc 1 // t1
brfalse $5
ldloc 1 // t1
stloc 0 // x
// demos typeswitch-sample cs(11 30   Console WriteLine(x)
ldloc 0 // x
call void [mscorlib]System Console   WriteLine (int32)
// demos/typeswitch-sample cs(11 42)  break
br $3
   x:\users dzh csharp compiler typeswitch-sample .. 'Fundamental) 45'
```

typeswitch-sample.il - Epsilon

File Edit Search Process Utility Window Help

```
./lcsc -visitor:Browser.BrowserVisitor,debug/Browser.dll hello.cs
./lcsc -visitor:bind -visitor:typecheck -visitor:Browser.BrowserVisitor,debu
./lcsc -visitor:typeswitch_source demos/typeswitch-sample.cs
   x:\users\drh\csharp\compiler\Inside.sh [Shell Fill] 66+
.maxstack 0
} // end of method Test.popsegment
.field static private class [mscorlib]System.Collections.ArrayList 'Symbols'
.method hidebysig public static void 'Foo'(object 'o') {
// demos/typeswitch-sample.cs(10,3): typeswitch (o)
.locals init ([0]object 't0')
ldarg 0 // o
stloc 0 // t0
// demos/typeswitch-sample.cs(11,3): case int32 (x):
ldloc 0 // t0
isinst int32
.locals init ([1]int32 't1')
stloc 1 // t1
ldloc 1 // t1
brfalse $5
ldloc 1 // t1
stloc 0 // x
// demos/typeswitch-sample.cs(11,20): Console.WriteLine(x)
ldloc 0 // x
call void [mscorlib]System.Console::'WriteLine'(int32)
// demos/typeswitch-sample.cs(11,42): break;
br $3
   x:\users\drh\csharp\compiler\typeswitch-sample.il [Fundamental] 45+
```

typeswitch-sample.il - Epsilon

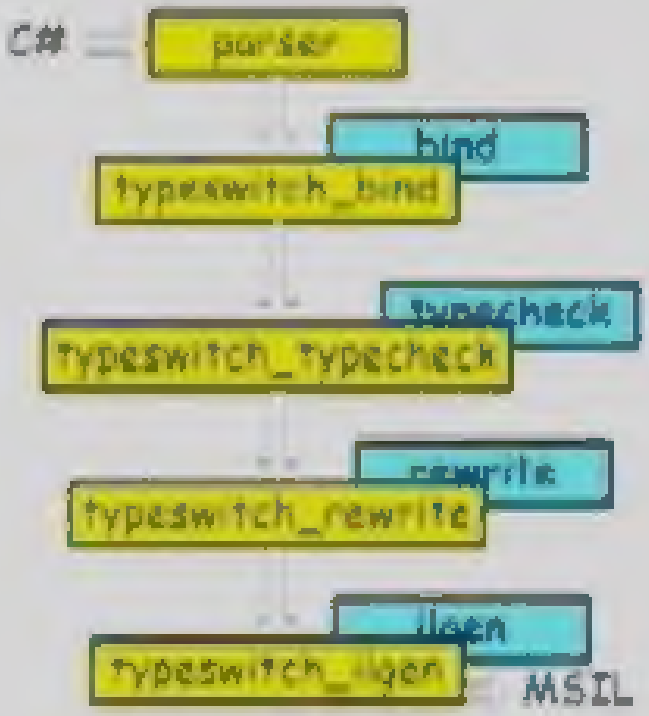
File Edit Search Process Utility Window Help

```
./lcsc -visitor:Browser.BrowserVisitor,debug/Browser.dll hello.cs
./lcsc -visitor:bind -visitor:typecheck -visitor:Browser.BrowserVisitor,debu
./lcsc -visitor:typeswitch_source demos/typeswitch-sample.cs
  x:\users\drh\csharp\compiler\Inside.sh [Shell Fill] 66%
.maxstack 0
} // end of method Test.popsegment
.field static private class [mscorlib]System.Collections.ArrayList 'Symbols'
.method hidebysig public static void 'Foo'(object 'o') {
// demos/typeswitch-sample.cs(10,3): typeswitch (o)
.locals init ([0]object 't
ldarg 0 // o
stloc 0 // t0
// demos/typeswitch-sample
ldloc 0 // t0
isinst int32
.locals init ([1]int32 't1
stloc 1 // t1
ldloc 1 // t1
brfalse $5
ldloc 1 // t1
stloc 0 // x
// demos/typeswitch-sample.cs(11,20): Console.WriteLine(x)
ldloc 0 // x
call void [mscorlib]System.Console::'WriteLine'(int32)
// demos/typeswitch-sample.cs(11,42): break;
br $3
  x:\users\drh\csharp\compiler\typeswitch-sample.il [Fundamental] 44%
```

PowerPoint Slide Show - [Inside.ppt]

The Full 9 Yards: Emitting MSIL

- Extend bind.cs (100 lines)
- Extend typecheck.cs (35 lines)
- Extend rewrite.cs (29 lines)
- Extend ilgen.cs (65 lines)
- (Demo)
- Alternative
  - Build assignment if-else ASTs directly, or rewrite AST as first visitor
  - Use bind.cs, typecheck.cs, rewrite.cs, ilgen.cs unmodified



Recap

- A simple C# compiler
  - Easy to understand (at least relatively)
  - Easy to modify
  - Ideal for language design experiments, language-based tools
- More info
  - http://manweb/pls/csharp/compiler/
  - Source Depot port MSR5010, //pls/csharp/compiler/
  - Current snapshot in \\research\Root\Public\drh\csharp\compiler